DOCUMENT RESUME

ED 233 517 EC 160 038

AUTHOR Rose, Susan; And Others
TITLE Computer Graphics and Creativity/Problem Solving Skills with Deaf and Severely Language Disordered Students: Parts I, II, and III.
PUB DATE Mar 83
NOTE 28p.; Paper presented at the Council for Exceptional Children Topical Conference on the Use of Microcomputers in Special Education (Hartford, CT, March 10-12, 1983).
PUB TYPE Speeches/Conference Papers (150) -- Guides - Classroom Use - Guides (For Teachers) (052)

EDRS PRICE MF01/PC02 Plus Postage.
DESCRIPTORS *Computer Assisted Instruction; *Computer Graphics; Computer Programs; *Creativity; Deafness; *Hearing Impairments; *Nonverbal Learning; *Problem Solving; Visual Learning

ABSTRACT

Three papers focus on applications of computer graphics with deaf and severely language impaired children. The first describes a drawing tablet software that allowed students to use visual and manipulative characteristics to enhance problem solving and creativity skills. Students were thus able to solve problems without the obstacles of language. The second article goes into greater detail about the software, with examples given for tasks of pattern recognition, numeric matrix, coding and decoding, and verbal activities. The software is designed to allow teachers to create graphic symbols for a variety of learning tasks. The final article describes a program in which a microcomputer was used to create visual imagery with children 8-15 years old. Operation requirements for the program are detailed. (CL)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED233517

Computer Graphics and Creativity/Problem Solving Skills with Deaf and Severely Language Disordered Students Parts I, II, and III.

Abstract

Deaf children rely extensively on their visual sense in acquiring information. Yet, most instructional programs for hearing impaired children regard verbal comprehension and English language proficiency as a major educational priority for this type of student. Our research examines the impact of nonverbal instruction on the hearing impaired, in particular, creativity and problem solving tasks with deaf children.

Present advances in technology have provided a new tool that has enormous potential in nonverbal instruction. Interactive computer graphics and appropriately designed educational software can facilitate individual problem solving tasks and related software as well as tools for expressing creativity through interactive computer graphics. Results of the pilot study with different hearing impaired students are discussed.

Susan Rose, Ph.D., Assistant Professor, Exceptional Children

Manjula B. Waldron, Ph.D., Assistant Professor, Engineering Graphics

William J. Kolomyjec, Ph.D., Assistant Professor, Engineering Graphics

The Ohio State University
Columbus, Ohio 43210

A Paper Presented at the Council for Exceptional Children's National Conference on the Use of Microcomputers in Special Education

Hartford, Connecticut
March 10-12, 1983

EC160038

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)

☒ This document has been reproduced as received from the person or organization originating it.

☐ Minor changes have been made to improve reproduction quality.

• Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

"PERMISSION TO REPRODUCE THIS MATERIAL HAS BEEN GRANTED BY

Susan Rose

TO THE EDUCATIONAL RESOURCES INFORMATION CENTER (ERIC)."

Computer Graphics and Creativity/Problem Solving Skills with Deaf and Severely Language Disordered Students: PartI

Susan Rose
Manjula Waldron
William Kolomyjec

The Ohio State University
Columbus, Ohio

When confronted with a problem we often invest time in thinking about it rather than actually solving it. Many times the solution just happens, the "Aha" reaction! At other times one needs to systematize the solution referring to prior knowledge and experience. Some problems require creative solutions, others are more mundane and routine. Whatever the situation, the problem solver needs to develop a schemata that helps him or her attack a problem in the generic sense. Good problem solvers are characterized by their generality and independence of subject matter while using the basic problem solving tools like inference, classification, subgoals, and contradictions (Wickelgren, 1974).

Shouksmith (1970) reported that intelligence and cognitive style as well as the ability to solve problems are involved in the creative thinking process. Persons valued as creative are those who are able to solve problems requiring the manipulation of information.

Deaf and language disordered students are frequently identified as lacking problem solving and creative thinking skills. It was hypothesized by the authors that this 'lack of' problem solving and creativity skills was perhaps an inaccessability to the problem or the situation due to the barriers imposed by the English language. An example of this situation is the student who is very capable with mathematical computations, such as $6 \div 2 = 3$, however, when given a problem, "Six cookies for two students, how many will each student get?" he

or she is unable to respond to the problem due to the language structure surrounding the components of the problem. Thus, in order to provide hearing impaired and language disordered students with problem solving experiences, a medium which was not bound by the English language needed to be developed.

Since hearing impaired persons' nonimpaired information access system is primarily visual, it was concluded that graphical representation can play an important role in acquisition of these skills (Waldron & Rose, 1983). Computer graphics, due to its highly visual and manipulative characteristics was selected as the medium to ascertain the problem solving abilities and creativity skills of hearing impaired learners.

A drawing tablet software allowed students to utilize the basic elements of any artwork: points, lines, rectangles, polygons, rotation of rectangles, filling in closed figures, and fitting a curve through a set of points. The students could erase the last step, the screen, or save the image. All this was done by hitting special keys on the keyboard. The users became familiar with various key functions through an illustrated manual. Each key with its function was introduced in a single step progression without adult intervention and without the use of the English language. When the students completed the elements presented in the manual, they were free to create their own images.

In a separate set of tasks, problem solving software was developed by the authors in the areas of pattern recognition, coding, tanagrams and numerical matrices. Each of the tasks used a graphic format with a multiple choice reponse mode. Students selected the solution or correct answer through a cursor control. The original software was created on a PDP 11/44 mini-computer and has since been adapted for use on the Apple II and on Apple IIE microcomputer.

The software design concentrated on providing the language impaired student with a creative medium, that of the computer and a few key controls. Students were in control of the elements or tools for creating or problem solving and used them. Since no specific directions were provided, each student had to utilize inference and deductive reasoning skills to discover the rules and select an answer. The uniqueness of the design was that no English words were used in the entire process. The system was demonstrated to a group of randomly selected students who attended a local hearing impaired program. They were essentially allowed to be on their own. The only intervention was if the student had a question or the system crashed. In these instances, monitors reset the program for the students.

The information acquired through this project is particularly valuable to teachers of language and/or hearing impaired students, to researchers and microcomputer users alike. Hearing impaired students were able to solve problems given a medium that was accessible, that is, visual, and a set of rules that were consistent. The students acquired the rules through experimentation and deduction. Problem solving and creative computer drawing tasks were highly motivating and challenging. This was particularly evident in the successful performance of eight and nine year old deaf children who were identified as behavior problems, hyperactive and multiply handicapped. These children attended to the computer graphics' tasks for ninety minutes without a break, excitedly sharing their creations and solutions with their neighbors.

Perhaps the most significant result was the fact that the students were able to find solutions to problems utilizing their personal schemata of deduction and inferences unencumbered by the English language system.

In addition, students with a 'tradition' of failure experienced success and hopefully, a feeling of self-worth.

Discussion

Technology in the special education classroom is neither new nor startling. Then what is all the excitement about? Why is there a National Conference, and why special publications? The answer appears to be embedded in the fact that never before has a technology been so powerful and so accessible to so many. The potential of the microcomputer as a significant teaching/learning tool can only be realized through a masterful teacher. The microcomputer can not teach. Software cannot make students learn. It is the careful coupling of individual student's learning skills, needs, and strengths with software, programs, and experiences that will utilize and maximize the potential of the technology of the microcomputer. The teacher holds the key to a world of learning in the computer age. This project has opened only one of the many avenues available to the hearing/language impaired learner through computer graphics in education.

References

Polya, G. How to solve it. Garden City, New York: Double Day & Company, 1957.

Shouksmith, G. Intelligence, creativity & cognitive style. New York: John Wiley & Son, 1970.

Waldron, M. & Rose, S. Visual thinking for severe language handicapped children through the use of computer graphics. Journal of Computer-based Instruction. May 1983, Vol. 9, Special Issue, p. 206-210.

Wickelgren, W. How to solve problems. San Fransisco, CA: W.H. Freeman Co., 1974.

Computer Graphics and Creativity/Problem Solving Skills with Deaf and Severely Language Disordered Students: Part II

by

M. B. Waldron, S. Rose and F. Barber

"Man need not bend to linguistic circumstance but may easily bend language to his needs." Joshua Whatmough

Software like LOGO allows the students to write their own programs interactively to express visually difficult concepts (Pappert, 1980). However, teachers find these programs hard to use developing thinking and problem solving skills; specifically, if the teaching methods involve testing specific skills and measuring progress. The software presented here recognizes this need of the classroom teacher and is designed so that, in addition to drawing pictures simply with a cursor and key control, the teachers can also design tasks in specific categories.

The two distinct advantages of our design are: 1) The software is user oriented; the user does not need to know any system software in order to generate their own graphic symbols. 2) The entire design of the software is modular in nature, hence the user can develop any task they want using their own symbol library. The task building program accesses the procedures it needs. Hence each developed task designed by the teacher can be stored on a separate diskette, independent of the master program used to generate the symbols.

User Oriented Software:

It was recognized that the software design would require sophisticated

programming. However, with some forethought and procedural planning one could write software that was user transparent. With this approach the teacher and students would not need to know the microcomputer programming language; rather, they need to know their own tasks, which they can "write" into the computer. PASCAL language was chosen for its wide availability on microcomputers, its ease in writing large programs in simple procedures, and good file handling capabilities. PASCAL allows the programs to be written with structure and hence has all the advantages of structural and procedural programming language. The other great advantage is the use of turtle graphics, which is easy to use in the programs to generate graphics and use these in any procedures (Luehrmann & Peckham, 1981). In this section the program design is presented along with a sample program for one of the problem solving tasks; other programs which are similar in design are mentioned. All programs are well documented and complete.

The problem solving and drawing software require a 64K RAM, APPLE II+ microcomputer, joystick and PASCAL language system diskette. The software described here allows the teacher to design problems and tasks for students in the area of pattern recognition, numerical matrix, graphical coding, and verbal tasks without knowing PASCAL, the system programming language. Built-in utilities and softwares are transparent to the user who only communicates in English and draws using the "cursor" via the control of the joystick and some selected keys on the keyboard. The system can be booted up and then the programs run, however, for ease of use the relevant files were put on the diskette so one could run the program by simply turning the power on (refer to APPLEPASCAL System Language Manual).

All graphical symbols were generated through the use of a program called "MAKESYMBOL". Figure 1 shows the menu used in designing any graphic symbol to

be used in tasks. These symbols could be shown in the square grid in any "color". The symbols could be "saved" and "recalled" by their name, see Figure 2. Their position could be moved in the x-y direction. They could be rotated, scaled, or erased in case an error was made. Thus a library of graphical symbols can be created by the teacher and then used in any of the tasks to be designed. In the following four tasks, the screen is divided into two parts, left for questions and right for answers.

1. Pattern Recognition: A graphical pattern consisting of up to six symbols is presented on the left; four possible answers are provided on the right. The student selects one of the four possible answers. If the answer is correct, they are rewarded by a symbol. If incorrect, nothing happens. See Figure 3.
2. Numeric Matrix: Here a matrix with user defined numbers is presented on the left. One of the boxes in the matrix is left blank. Based on the given numbers, the student guesses the underlying operation of the matrix and chooses the answer from four possible answers which fit in the blank box. If the answer is correct, a reward symbol appears; otherwise, nothing happens.
3. Coding - Decoding Tasks: In this program, two sets of nine shapes appear on the top of the screen with each shape is associated a symbol. Four shapes appear on the left. The student must choose the associated symbol patterns from four possible choices on the right hand side of the screen. If the choice is correct, a reward symbol appears.
4. Verbal tasks: A verbal task presents a question written on the left side of the screen. The student selects one of the four possible answers, and if the answer is correct, a reward symbol appears.

In all the tasks described above, the teacher has full control of what graphic, number, or letters to put in the question space or the answer space. The process of building and running the task is done by two major sets of programs. To build the tasks, DATA/BUILDTASK diskette is put in Drive 1, which contains the symbols and the programs. The BDTASK diskette goes in Drive 2 where the designed tasks are stored. After the tasks are built and control RUN file is written on the task diskette, the task can be run by putting the task diskette in Drive 1 and rebooting the system.

1. Building tasks: Involves a three step procedure. Turn power off. Put DATA/BUILDTAKS diskette in Drive 1. Put the task diskette in Drive 2 and turn the power on. The program will then run. A build task menu appears to allow you to select the appropriate task you want to build. Up to ten tasks may be put on a diskette.
2. Running tasks: This also requires three steps. Turn power off. Put the task diskette in Drive 1. Put any diskette with Question.DATA, Answer.DATA and Yes.DATA and any other Symbol.DATA file on it in Drive 2 and turn the power on. The program will then ask for the student information. Then the program will run all the prepared tasks. After the last question has been answered, subject information prompt will come back. The student information is stored in a results file to show the teacher how each of the students performed.

A sample build and run program for pattern recognition is given in Appendix 1. The program is well documented, explaining all the variables and procedures used. It also contains the explanation of use so the reader can select the relevant portion of the program for inspection.

Discussion:

The user friendly software was designed so that teachers could create graphic symbols and use them in problem solving and thinking tasks of different sorts. The software design allows teachers to generate symbols, store them, and use them in the tasks where most suited. The program "MAKESYMBOL" allows the student and teacher alike to draw any designs and store them for future use, thus providing a creative flexible medium to the hearing impaired student and his/her teacher. The main use envisaged for this software is to allow teachers to generate low verbal highly graphical tasks to encourage visual, deductive, and inductive thinking and problem solving skills of these students. It is hoped that the ease of use of software to design complex visual problems will encourage teachers to use microcomputer graphics in a more constructive manner than simple drill and practice. The teacher prepares material no different than on paper; however, the students work both interactively and independently, and have an opportunity to explore their own thinking and creative skills with immediate feedback.

References:

Luehrman, A. & Peckham, H. Apple Pascal: A Hands On Approach. New York: McGraw-Hill, 1981.

Pappert, S. Mindstorms New York: Basic Books, Inc., 1980.

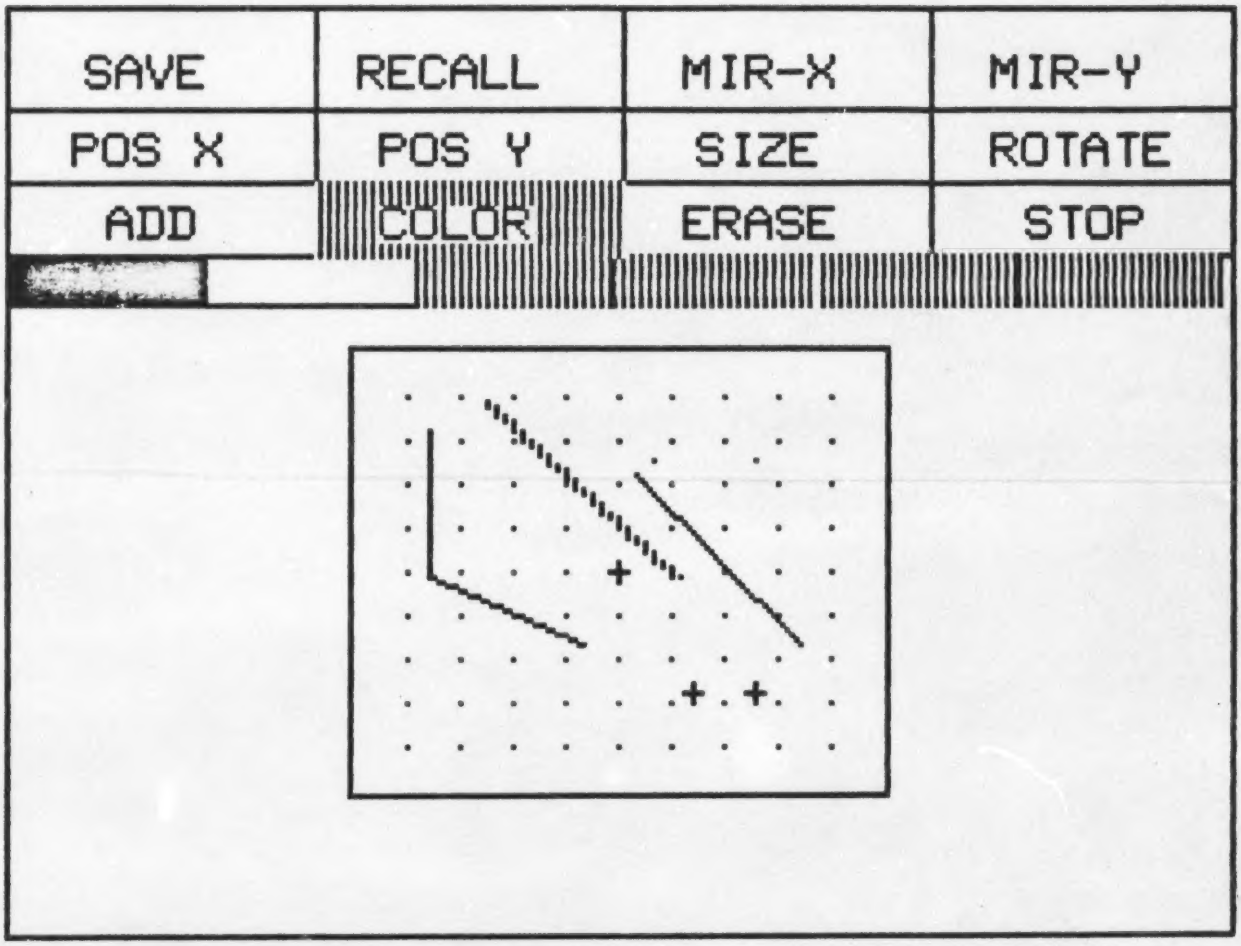

Figure 1: Make symbol menu

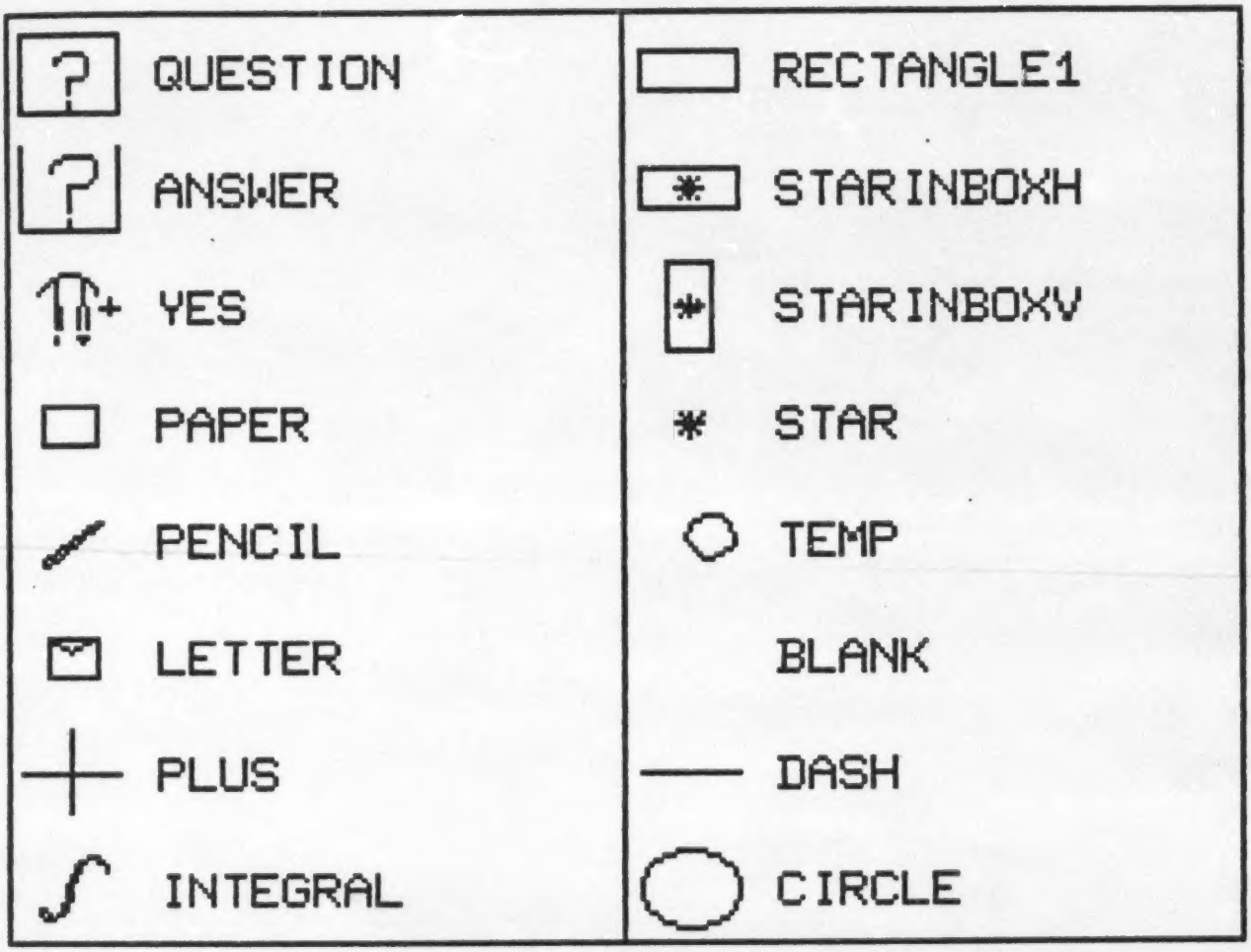

Figure 2: Symbols stored

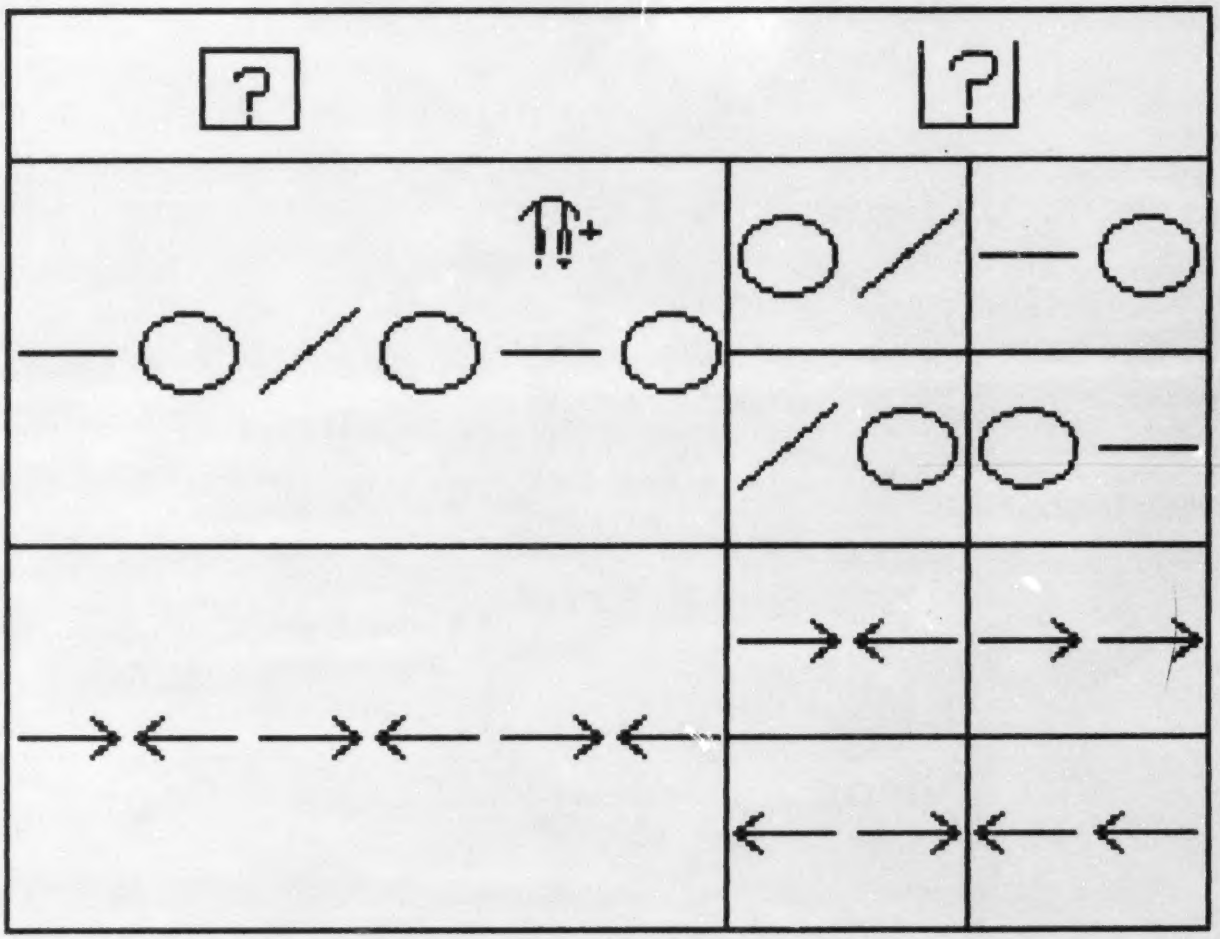

Figure 3: Pattern recognition task

Computer Graphics and Creativity / Problem Solving Skill with Deaf and Severely Language Disordered Students.

Part III: Using the Microcomputer as a Creative Tool

by

William J. Kolomyjec, Ph.D, M.F.A

in conjunction with

Drs. S. Rose and M. B. Waldron

Abstract

This is the final part in a series of three articles having to do with exploring Creativity through Computer Graphics with Hearing Impaired Children. The body of this article consists of a discussion of an Applesoft (TM) Basic program which will allow an individual using an Apple (TM) microcomputer to create visual imagery. The software presented illustrates how computer graphics might become a medium allowing individual expression. It is based on a research instrument originally designed for use with hearing impaired subjects. Children ages 8 to 15 produced imagery with similar software which, in turn, provided a means to study visual creativity.

Introduction

In the original research project ('1') the software was written in FORTRAN on a DEC PDP 11/44 minicomputer. See Part 1 of this trilogy. Timesharing was used to support 10 graphics terminals. There were many advantages for conducting the experiment in this

manner, vis-a-vis, high information exchange rate, simultaneous availability of a large number of terminals, mass storage and hardcopy. Moreover, experimental conditions could be easily maintained. However, there are disadvantages to a large system approach. Namely, access to the hardware is usually restricted. Subjects have to be transported to the facility. The software is not transportable, i.e., it cannot be used on other systems. Because the experience was very positive for both the subjects and the researchers, the next logical step seemed to adapt the software to the microcomputer. On small, user-oriented machines more individuals can use this instrument for both research and nonresearch activities. The software has enormous recreational value as well as the potential to allow individual expression. A brief discussion of the program follows.

Hardware Requirements

A standard Apple II Plus (TM) with 48K RAM or Apple IIe is required. The program generates monochromatic images and therefore does not require a color monitor. A joystick with two buttons or both game paddles is necessary.

Operation

Upon running the program a text menu is displayed. The following FUNCTION OPTIONS are provided:

```
0 = ERASE SCREEN
1 = LINE (END POINTS)
2 = LINE (CONTINOUS)
3 = RECTANGLES (OPPOSITE CORNERS)
4 = CIRCLE (CENTER  RADIUS)
5 = ARC (THRU THREE POINTS)
6 = SPLINE (UP TO 15 POINTS)
7 = EXIT
```

Directions for use: The user selects an option via the keyboard. Only option 0 erases the screen. Entering a drawing mode, numbers 1 - 6, will not clear the screen. Option 7 will exit the program.

In each drawing mode (options 1 - 6) depressing the paddle button 0 will perform the function, depressing paddle button 1 will return to the menu. Spline is an exception. In the spline mode (option 6) pushing button 0 will define the curve, pushing button 1 will draw it. This mode is not exited until button 1 is pushed again, after the curve is complete. That's all there is to it!

Program Description

Main program.

The main program, lines 100-999, typifies good graphics style. Line 120 keeps the program above the graphics portion of memory ('2'). In line 125 a statement function is defined which effectively adjusts the screen origin and corrects the aspect ratio. Line 130 transfers control to a subroutine that performs initialization of the program variables, as well as the clipping and joystick routines. Line 140 causes the text menu to be displayed.

After the user selects a function, line 150, it is checked for correctness. At line 170 control is transfered via a multiple branch statement to the selected function's subroutine. Branching to each function entails: 1) Switching into the graphics mode without erasure. 2) Executing the proper code to perform the function. And, 3) Upon return, redisplaying the menu. The program again awaits user input. The exceptions to this process are erase and exit.

Subroutines

PLOTSUBC, Lines 1000-1210 contain the code for a plotting subroutine with clipping. This subroutine is used by the circle, arc and spline functions because they have the potential to, in part, lie outside the screen boundaries. The clipping algorithm significantly reduces the plotting speed of the microcomputer due to the extensive number calculations it is required to perform.

DON'T ERASE, lines 1250-1260. A series of "soft switches" are POKE'd to re-enter the graphics mode without clearing the present contents of screen memory ('3').

ARC SUB, lines 1700-1860. This is a general arc drawing subroutine that requires six parameters: The X and Y coordinates of the center and a radius account for three parameters. A beginning and final angle (in degrees) and a flag to indicate drawing direction, clockwise or counter clockwise constitute the remaining parameters. The smoothness of the arc, i.e., the sampling rate, varies with the size of its radius.

LINE1, lines 3000-3080. This is the first function option subroutine. The cursor is displayed (line 3010). Locating the cursor and depressing button 0 will continuously define line segment end points. After the first end point is entered that pixel (discrete screen coordinate) is illuminated. After the other endpoint is selected the line segment is drawn. This process will continue until button 1 is depressed, returning to the menu.

LINE2, lines 3500-3590. This function works in a manner similar to LINE1 with the exception that every time a location is selected with button 0 after the second location, a line segment is connected to the previous endpoint. Thus, a continuous line is produced. Again, button 1 exits the routine.

CURSOR RECTANGLES, lines 4000-4120. The opposite corners of a rectangle are defined by positioning the cursor and depressing button 0. These locations are momentarily displayed. The computer checks the the coordinate data of these locations and calculates the width and height of the desired rectangle then draws it. This process will continue until button 1 is depressed.

CIRCLE, lines 4500-4620. This function mode uses the ARC SUB. To use the arc subroutine six variables must be defined. Since a complete circle is required the beginning and final angle and direction are constant. The remaining variables for center and radius can be determined by entering two points using the cursor routine. The first point is the center and the second point is a

point on the circle's circumference. If part of the selected circle exceeds any screen boundary then PLOTSUBC will clip off the overlapping portion.

ARC THRU THREE POINTS, lines 5000-5220. Three points are entered using the cursor. The computer will fit an arc through the points in the order in which they were entered. Try not to place the three points along a straight line because this doesn't make much sense and it might confuse the algorithm. This routine requires the support of two other subroutines: COMPUTE CENTER AND RADIUS and THETA.

SPLINE, lines 5500-5670 and CURVE COORD. FINDER, lines 6500-6560. These two subroutines work together to produce a spline curve through a series of points. When this function is selected the user may locate and enter, using button 0, up to fifteen (15) points on the screen. Each point or pixel will be illuminated as selected. Depressing button 1 will tell the computer to fit the spline through these points. Unfortunately the process is slow due to the calculations involved. After the spline curve is drawn, and unless button 1 is depressed, the process is repeated.

IMPROVED CURSORSUB2, lines 7000-7090. This subroutine lies at the heart of the interactive capabilities of this program. The cursor, which is a high resolution "shape," is controlled on the screen by the analog inputs (joystick or paddle controls). The location of the center of the cursor at any time it appears on the screen is contained by the integer variables X%,Y%. When either paddle button is depressed this screen location is noted

and also which button was depressed. Thus this information can be used by the various routines. This cursor routine is "improved" because it will not allow data to be read in until the button is released and depressed again. See the Applesoft (TM) reference manual ('4') or the Apple II User's Guide ('5') to learn more about XDRAW.

INITCURSORSUB, lines 7500-7560. This routine is only used once. It is called by the initialization subroutine when the program is first executed. It POKE's the shape (the cross shaped cursor) into an obsure part of memory ($300-$30F). This shape can then be used by the XDRAW predefined function in the cursor subroutine.

COMPUTE CENTER AND RADIUS, lines 8500-8570. This subroutine computes the center and radius of a circle given three points in two dimensional space. It is obviously used by the ARC THRU THREE POINTS subroutine.

THETA, lines 8750-8850 simulates a function which returns the angle (in radians) formed between a point in a plane and the +X axis of the coordinate system.

TEXT MENU, lines 9000-9110. The main menu is produced by this subroutine. When it is executed it clears the text screen and prints the option menu. In other versions of this program this routine is replaced by a cursor driven menu where the cursor's position is used to select a function option. This version will not be given.

CLIPPING INITIALIZATION, lines 10000-10120. Like INITCURSORSUB this routine is only called once. An array is filled with the appropriate value that defines the entire display screen to be a "viewport" for the clipping algorithm.

One final word about the listing that follows. In order to speed up program execution and fit it into a 48K machine the program had to be compressed. The majority of the REM statements (comments) were removed and the technique of using the delimiter (:) such that multiple statements could be placed on one line was employed ('6'). Although these techniques expedite execution and save space, they also make the listing hard to read. For this, please accept an apology. Lastly, execution time can be significantly improved by compiling this program using any of several commercially available compilers. However these are often expensive to purchase and not very easy to use.

Conclusion

Software is what gives intelligence to a computer enabling it to function as an instrument or tool. The program discussed above and given below will transform the Apple (TM) into an image generating tool which can be used in a variety of educational applications. It is particularly useful for handicapped individuals in that it only requires (in this version) touching a number key, moving a joystick and depressing paddle buttons. As was suggested, keyboard input can be eliminated by replacing the text menu with a graphics menu. Thus

only a joystick and two buttons would be required. In our research we have demonstrated the utility of this instrument. By publishing this software we make this tool available to any interested person willing to take the time to enter it into their own Apple (TM).

References

1. Kolomyjec, W.J, Rose, S. and Waldron, M. B., "Creativity through Computer Graphics with Hearing Impared Children." Final Report for Spencer Foundation Young Scholars Grant, Department of Educational Research. The Ohio State University.: June 1982.

2. Poole, L. et al. Apple II User's Guide. Osborne/McGraw Hill. Berkeley. CA.: 1981. p. 204.

3. Ibid. p. 205.

4. Applesoft: Basic Programming Reference Manual. Apple Computer Company. (Product No. A210006). Cupertino, CA.: 1981. p. 98.

5. Poole, L. et al., Ibid. p. 312.

6. Applesoft: Basic Programming Reference Manual. Apple Computer Company. (Product No. A210006). Cupertino, CA.: 1981. p. 118-120.

```
]?CHR$(9);"80N":LIST

100  REM  <<<<< MICROCOMPUTER DRAWING >>>>>
101  REM
102  REM  COPYRIGHT 1983 W.J.KOLOMYJEC
103  REM
104  REM  <<<<< TEXT MENU VERSION >>>>>
105  REM
120  LOMEM: 24576
125  DEF  FN Y(Y) = 192 - (Y * 0.881 + 0.5)
130  GOSUB 8000
140  GOSUB 9000
150  PRINT : PRINT "ENTER OPTION : ": GET O$: PRINT O$
155 O% =  VAL (O$)
160  IF O% < 0 OR O% > 7 THEN 140
170  ON O% + 1 GOTO 180,190,200,210,220,230,240,250
180  HGR2 : GOTO 140: REM  ERASE
190  GOSUB 1250: GOSUB 3000: GOTO 140
200  GOSUB 1250: GOSUB 3500: GOTO 140
210  GOSUB 1250: GOSUB 4000: GOTO 140
220  GOSUB 1250: GOSUB 4500: GOTO 140
230  GOSUB 1250: GOSUB 5000: GOTO 140
240  GOSUB 1250: GOSUB 5500: GOTO 140
250  TEXT : HOME : VTAB 10: PRINT "END COMPUTER DRAWING..."
999  END
1000  REM  <<<<< PLOTSUBC >>>>>
1010  IF P = 2 THEN X1 = X:Y1 = Y: RETURN
1020  IF P = 1 THEN 1040
1030  PRINT : PRINT "PEN ERROR, P NOT 1 OR 2": STOP
1040 X2 = X:Y2 = Y
1050  FOR I = 0 TO NW - 2
1060 V1 = (W(I + 1,1) - W(I,1)) * X1 + (W(I,0) - W(I + 1,0)) * Y1 + W(I + 1,0) *
 W(I,1) - W(I,0) * W(I + 1,1)
1070 V2 = (W(I + 1,1) - W(I,1)) * X2 + (W(I,0) - W(I + 1,0)) * Y2 + W(I + 1,0) *
 W(I,1) - W(I,0) * W(I + 1,1)
1075  REM  ADD 0.0001 TO THE DIVISOR TO PREVENT DIVIDE BY ZERO ERRORS
1080 PT = V1 / ((V1 - V2) + 0.0001)
1090  IF V1 > 0 THEN 1140
1100  IF V2 < 0 THEN 1200
1110 X1 = X1 + PT * (X2 - X1)
1120 Y1 = Y1 + PT * (Y2 - Y1)
1130  GOTO 1170
1140  IF V2 > 0 THEN 1170
1150 X2 = X1 + PT * (X2 - X1)
1160 Y2 = Y1 + PT * (Y2 - Y1)
1170  NEXT I
1180 Y1 = 192 - (Y1 * AR + 0.5):Y2 = 192 - (Y2 * AR + 0.5)
1190  HPLOT X1,Y1 TO X2,Y2
1200 X1 = X:Y1 = Y
1210  RETURN
1250  REM  <<<<< DON'T ERASE >>>>>
1260  POKE 49232,0: POKE 49239,0: POKE 49237,0: RETURN
1700  REM  <<<<< ARC SUB >>>>>
1710  IF R <  = 8 THEN NS = 10: GOTO 1760
1720  IF R >  = 9 AND R < 16 THEN NS = 20: GOTO 1760
1730  IF R >  = 17 AND R < 32 THEN NS = 30: GOTO 1760
1740  IF R >  = 33 AND R < 64 THEN NS = 45: GOTO 1760
1750  IF R >  = 65 THEN NS = 90
```

```
1770 F = FANG / 57.295779:L = LANG / 57.295779
1780  FOR J = 1 TO N
1790 PCT = (J - 1) / (N - 1 + 0.00001)
1800  IF IDIR% < 0 THEN PCT = 1.0 - PCT
1810 AN = (L - F) * PCT + F
1820 X =  INT (R *  COS (AN) + XCT + 0.5):Y =  INT (R *  SIN (AN) + YCT + 0.5)
1830  IF J <  > 1 THEN 1850
1840 P = 2: GOSUB 1000: GOTO 1860
1850 P = 1: GOSUB 1000
1860  NEXT J: RETURN
3000  REM  <<<<< LINE1 >>>>>
3010  GOSUB 7000: REM  CURSOR
3020  IF F1% = 1 THEN 3080
3030 X1 = X%:Y1 = Y%: HPLOT X1,Y1
3040  GOSUB 7000
3050  IF F1% = 1 THEN 3080
3060 X2 = X%:Y2 = Y%: HPLOT X1,Y1 TO X2,Y2
3070  GOTO 3010
3080  RETURN
3500  REM  <<<<< LINE2 >>>>>
3510  GOSUB 7000
3520  IF F1% = 1 THEN 3590
3530 X1 = X%:Y1 = Y%: HPLOT X1,Y1
3550  GOSUB 7000
3560  IF F1% = 1 THEN 3590
3570 X2 = X%:Y2 = Y%: HPLOT X1,Y1 TO X2,Y2
3580 X1 = X2:Y1 = Y2: GOTO 3550
3590  RETURN
4000  REM  <<<<< CURSOR RECTANGLES >>>>>
4010  GOSUB 7000
4020  IF F1% = 1 THEN 4120
4030 XR = X%:YR = Y%: HPLOT XR,YR
4040  GOSUB 7000
4050  IF F1% = 1 THEN 4120
4060 XS = X%:YS = Y%: HPLOT XS,YS
4070  IF XS < XR THEN T = XR:XR = XS:XS = T
4080  IF YS < YR THEN T = YR:YR = YS:YS = T
4090 H =  ABS (YS - YR):W =  ABS (XS - XR)
4100  HPLOT XR,YR TO XR + W,YR TO XR + W,YR + H TO XR,YR + H TO XR,YR
4110  GOTO 4010
4120  RETURN
4500  REM  <<<<< CIRCLE >>>>>
4510 FANG = 0:LANG = 360:IDIR% = 1
4520  GOSUB 7000
4530  IF F1% = 1 THEN 4620
4540 X1 = X%:Y1 = YS%: HPLOT X1, FN Y(Y1)
4550  GOSUB 7000
4560  IF F1% = 1 THEN 4620
4570 X2 = X%:Y2 = YS%: HPLOT X2, FN Y(Y2)
4590  HCOLOR= 4: HPLOT X1, FN Y(Y1): HCOLOR= 3
4600 XD =  ABS (X2 - X1):YD =  ABS (Y2 - Y1):R =  SQR (XD * XD + YD * YD)
4610 XCT = X1:YCT = Y1: GOSUB 1700: GOTO 4520
4620  RETURN
5000  REM  <<<<< ARC THRU THREE POINTS >>>>>
5010  GOSUB 7000
5020  IF F1% = 1 THEN 5220
5030 X1 = X%:Y1 = YS%: HPLOT X1, FN Y(Y1)
5040  GOSUB 7000
5050  IF F1% = 1 THEN 5220
5060 X2 = X%:Y2 = YS%: HPLOT X2, FN Y(Y2)
5070  GOSUB 7000
5080  IF F1% = 1 THEN 5220
5090 X3 = X%:Y3 = YS%: HPLOT X3, FN Y(Y3)
5100  GOSUB 8500: REM  CALC XO,YO,R
5110 XCT = XO:YCT = YO:XX = X1 - XCT:YY = Y1 - YCT: GOSUB 8750:FANG = THETA * 57
```

```
5120 XX = X2 - XCT:YY = Y2 - YCT: GOSUB 8750:MANG = THETA * 57.295779
5130 XX = X3 - XCT:YY = Y3 - YCT: GOSUB 8750:LANG = THETA * 57.295779
5140 IF FANG < MANG AND MANG < LANG THEN IDIR% = 1: GOTO 5200
5150  IF FANG < LANG AND LANG < MANG THEN IDIR% =  - 1:TEMP = LANG:LANG = FANG +
 360:FANG = TEMP: GOTO 5200
5160  IF MANG < FANG AND FANG < LANG THEN IDIR% =  - 1:TEMP = LANG:LANG = FANG +
 360:MANG = MANG + 360:FANG = TEMP: GOTO 5200
5170  IF MANG < LANG AND LANG < FANG THEN IDIR% = 1:MANG = MANG + 360:LANG = LAN
G + 360: GOTO 5200
5180  IF LANG < FANG AND FANG < MANG THEN IDIR% = 1:LANG = LANG + 360: GOTO 5200

5190 IDIR% =  - 1:TEMP = LANG:LANG = FANG:FANG = TEMP
5200  GOSUB 1700
5210  GOTO 5010
5220  RETURN
5500  REM  <<<<< SPLINE >>>>>
5510 K = 1
5520  GOSUB 7000
5530  IF F1% = 1 THEN 5590
5540  HPLOT X%, FN Y(YS%)
5550 C(K,1) = X%:C(K,2) = YS%
5560 K = K + 1
5570  IF K > NP% THEN K = NP%: REM  LIMIT CHECK
5580  GOTO 5520
5590 KNT = K - 1:NS = IRS * KNT:P = 2
5600  FOR J = 1 TO NS
5610 T = (J - 1) / (NS - 1) * (KNT - 1) + 1
5620  GOSUB 6500
5630  IF J <  > 1 THEN P = 1
5640  GOSUB 1000
5650  NEXT J
5660  GOSUB 7000: IF F1% = 1 THEN  RETURN
5670 K = 1: GOTO 5540
6500  REM  <<<<< CURVE COORD FINDER >>>>>
6510 X = 0:Y = 0:D = 0:S =  - 1
6520  FOR I = 1 TO KNT
6530 F =  EXP (T - I + 0.001):F = 2 / (F - 1 / F)
6540 X = X + S * C(I,1) * F:Y = Y + S * C(I,2) * F
6550 D = D + S * F:S =  - S
6560  NEXT I:X = X / D:Y = Y / D: RETURN
7000  REM  <<<<< IMPROVED CURSOR2SUB >>>>>
7005  IF  PEEK (P0) < 127 AND  PEEK (P1) < 127 THEN FF% = 0: GOTO 7010
7006 FF% = 1
7010 XP =  PDL (0) / 255:YP =  PDL (1) / 255:X% = 279 * XP:Y% = 191 * YP: XDRAW
1 AT X%,Y%
7020 PX% = X%:PY% = Y%
7030 XP =  PDL (0) / 255:YP =  PDL (1) / 255:X% = 279 * XP:Y% = 191 * YP
7040  XDRAW 1 AT PX%,PY%: XDRAW 1 AT X%,Y%
7050  IF  PEEK (P0) > 127 THEN F1% = 0: GOTO 7070
7060  IF  PEEK (P1) > 127 THEN F1% = 1: GOTO 7070
7065 FF% = 0: GOTO 7020
7070  IF FF% = 1 THEN 7020
7075 FF% = 1
7080  XDRAW 1 AT X%,Y%
7090 YS% = 217 * (1 - YP): RETURN
7500  REM  <<<<< INITCURSORSUB >>>>>
7510  ROT= 0: SCALE= 2:LOOP = 6
7520  DATA  1,0,4,0,36,76,145,58
7530  DATA  63,63,23,82,9,36,4,0
7540  FOR J = 768 TO 783: READ D: POKE J,D: NEXT J
7550  POKE 232,0: POKE 233,3
7560  RETURN
8000  REM  <<<<< INIT PROGRAM VARIABLES >>>>>
8010  HGR : HGR2 : HCOLOR= 3
8020 DL = 50:P0 = 49249:P1 = 49250:F1% = 0
8030  GOSUB 7500: REM  INIT CURSOR
```

```
8040  GOSUB 10000: REM  CLIPPING INIT
8050 IRS = 5:NP% = 15: DIM C(NP%,2)
8060  RETURN
8500  REM  <<<<< COMPUTE CENTER & RADIUS >>>>>
8510 T1 = (Y2 - Y1) / (X2 - X1 + 0.00001):T2 = (Y3 - Y1) / (X3 - X1 + 0.00001)
8520 T3 = ((X2 - X1) * (X2 + X1)) + ((Y2 - Y1) * (Y2 + Y1))
8530 T4 = T3 / (2 * (X2 - X1) + 0.00001)
8540 T5 = ((X3 - X1) * (X3 + X1)) + ((Y3 - Y1) * (Y3 + Y1))
8550 T6 = T5 / (2 * (X3 - X1) + 0.00001)
8560 YO = (T6 - T4) / (T2 - T1):XO = T6 - (T2 * YO)
8570 R =  SQR ((X3 - XO) ^ 2 + (Y3 - YO) ^ 2): RETURN
8750  REM  <<<<< THETA FUNCTION >>>>>
8760 PI = 4 *  ATN (1.):UNDF = 0
8770  IF XX < 0 THEN THETA = PI +  ATN (YY / XX): RETURN
8780  IF XX = 0 THEN 8800
8790  GOTO 8830
8800  IF YY < 0 THEN THETA = PI * 1.5: RETURN
8810  IF YY = 0 THEN THETA = UNDF: RETURN
8820 THETA = PI / 2: RETURN
8830  IF YY < 0 THEN THETA = 2 * PI +  ATN (YY / XX)
8840  IF YY = 0 THEN THETA = 0: RETURN
8850 THETA =  ATN (YY / XX): RETURN
9000  REM  <<<<< TEXT MENU >>>>>
9010  HOME : TEXT : VTAB 5
9020  PRINT "FUNCTION OPTIONS:"
9030  PRINT : PRINT "   0=ERASE SCREEN"
9040  PRINT "    1=LINE (END POINTS)
9050  PRINT "    2=LINE (CONTINUOUS)"
9060  PRINT "    3=RECTANGLE (OPPOSITE CORNERS)"
9070  PRINT "    4=CIRCLE (CENTER & RADIUS)"
9080  PRINT "    5=ARC (THRU THREE POINTS)"
9090  PRINT "    6=SPLINE (UP TO 15 POINTS)"
9100  PRINT "    7=EXIT"
9110  RETURN
10000  REM  <<<<< CLIPPING INITIALIZATION >>>>>
10010 AR = 0.881: REM  ASPECT RATIO
10020 NW = 5: REM  NUMBER OF CONNECTED POINTS IN WINDOW
10030  DIM W(NW - 1,1)
10040  REM  RECTANGULAR WINDOW
10050 XL = 0:XR = 279
10060 YB = 0:YT = 217
10065  REM  EFFICIENT ARRAY USAGE
10070 W(0,0) = XL:W(0,1) = YB
10080 W(1,0) = XL:W(1,1) = YT
10090 W(2,0) = XR:W(2,1) = YT
10100 W(3,0) = XR:W(3,1) = YB
10110 W(4,0) = XL:W(4,1) = YB
10120  RETURN
22222  REM  <<<<< END OF MICROCOMPUTER DRAWING >>>>>

]PR#0
```